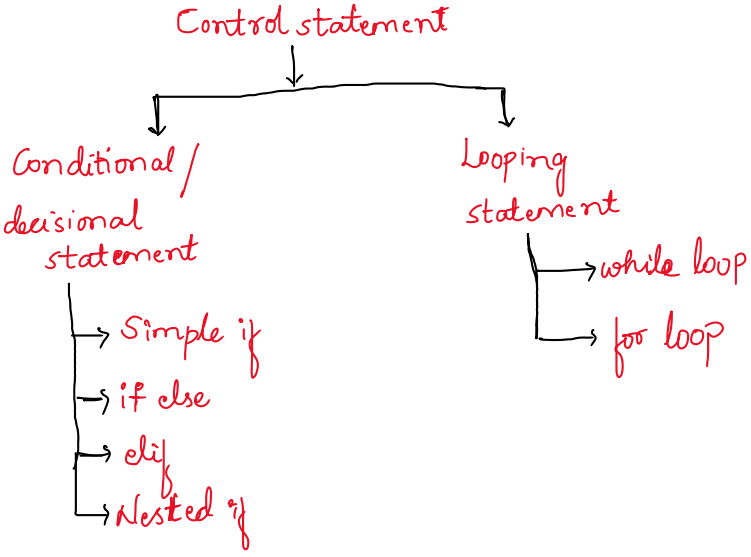
06 February 2025 08:13

Day-20

Control Statement:
--- It is used to control the flow of execution.

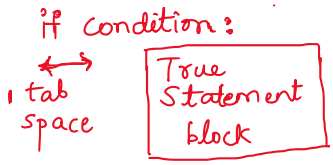
Types:



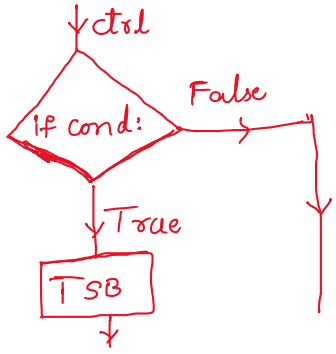
Conditional statement:
--- It is used to control the flow of execution based on conditions.

1) **Simple if:**
--- It is a keyword which is used to check the condition and it will execute the statement block if the condition is True. It will ignore the statement block if the condition is False.

Syntax:



Flow diagram:



Programs:

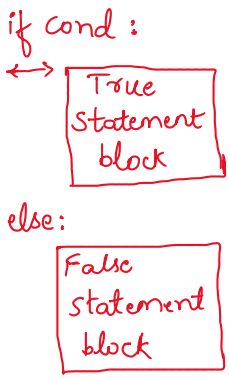
```
#Simple if

#WAP to check whether the number is even.
'''
num=int(input('Enter the number: '))
if num%2==0:
   print(num,'is even')'''

#WAP to check whether string has exactly 5 characters in it .
'''
s=input('Enter the string: ')
if len(s)==5:
   print(s,'has exactly 5 characters in it')'''
```

```
#WAP to check whether the number is greater than 200 .
'''
num=int(input('Enter the number: '))
if num>200:
   print(num,'is greater than 200')'''

#WAP to print the square of the number if the number is multiple of 3.
'''
num=int(input('Enter the number: '))
if num%3==0:
   print(num**2)'''

#WAP to check whether the character is Uppercase .
'''
s=input('Enter the character: ')
if 'A'<=s<='Z':
   print(s,'is uppercase')'''
```

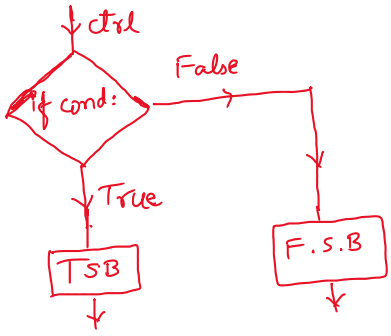
2) if else:

--- It is an advance version of simple if. It will execute the True statement block if the condition is True. It will execute the False statement block if the condition is False.

Syntax:



Flow diagram:



Programs:

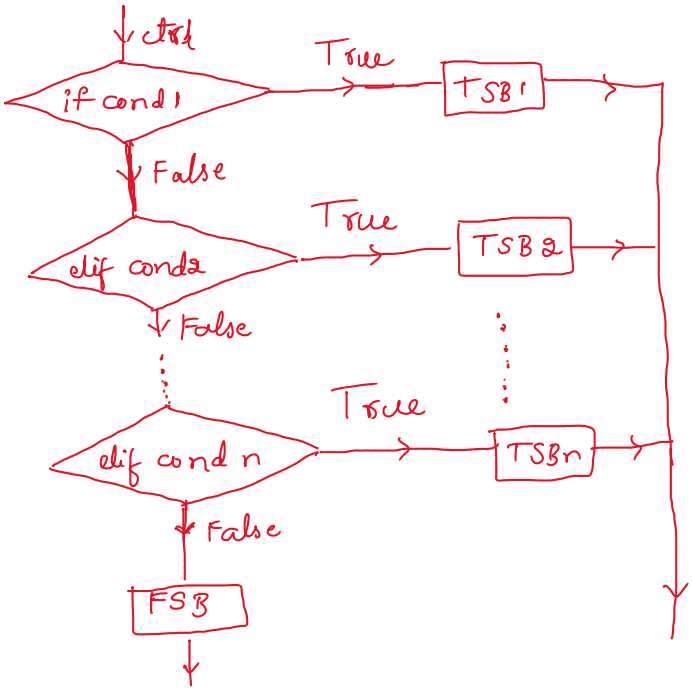
```
#WAP to check the given data is float or not.
'''
data=eval(input('Enter the data: '))
if type(data)==float:
   print('given data is float')
else:
   print('given data is not float')'''

#WAP to check whether the string is palindrome or not
'''
s=input('Enter the string: ')
if s==s[::-1]:
   print(s,'is palindrome')
else:
   print(s,'is not palindrome')'''

#WAP to check whether the given character is vowel or not.
'''
s=input('Enter the character: ')
if s in 'aeiouAEIOU':
   print(s,'is vowel')
else:
   print(s,'is not vowel')'''
```

```
#WAP to check whether the given data is single valued datatype or not.
'''
data=eval(input('Enter the data: '))
if type(data) in [int, float, complex, bool] :
   print(data,'is SVDT')
else:
   print(data,'is not SVDT')'''

'''
data=eval(input('Enter the data: '))
if type(data) == int or type(data) == float or  type(data) == complex or type(data) == bool:
   print(data,'is SVDT')
else:
   print(data,'is not SVDT')'''


#WAP to check whether the given integer is 3 digit number or not.
'''
num = abs(int(input('Enter the number: ')))
if 100<=num<=999:
   print(num, 'is 3 digit number')
else:
   print(num,'is not 3 digit number')'''
```

3) elif : Whenever we want to check multiple condition and execute multiple statement block of each and every condition then elif is used.

Syntax:

Flow Diagram:



Programs:

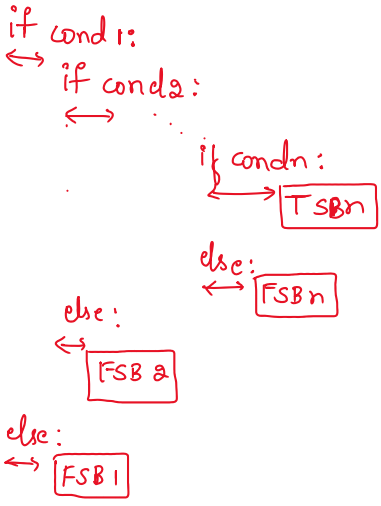
```
#elif

# WAP to find the relation between two integers.
'''
num1=int(input('Enter the number1: '))
num2=int(input('Enter the number2: '))
if num1>num2:
   print(num1,'is greater')
elif num1<num2:
```

```
    print(num1,'is lesser')
else:
    print(num1,'is equal to',num2)'''

# WAP to check whether the character is uppercase or lowercase or digit or special characters.
'''
ch=input('Enter the character: ')
if 'A'<=ch<='Z':
    print(ch,'is uppercase')
elif 'a'<=ch<='z':
    print(ch,'is lowercase')
elif '0'<=ch<='9':
    print(ch,'is digit')
else:
    print(ch,'is special character')'''

# WAP to check whether the given integer is single digit or two digit or three digit or more than 3 digit
'''
num=abs(int(input('Enter the number: ')))
if 0<=num<=9:
    print('single digit')
elif 10<=num<=99:
    print('two digit')
elif 100<=num<=999:
    print('three digit')
elif num>999:
    print('more than 3 digit')'''

# WAP to find the greatest among 4 numbers
'''
a,b,c,d=int(input('Enter the num1: ')),int(input('Enter the num2: ')),int(input('Enter the num3: ')),int(input('Enter the num4: '))
if a>b and a>c and a>d:
    print(a,'is greatest')
elif b>a and b>c and b>d:
    print(b,'is greatest')
elif c>a and c>b and c>d:
    print(c,'is greatest')
else:
    print(d,'is greatest')'''
```

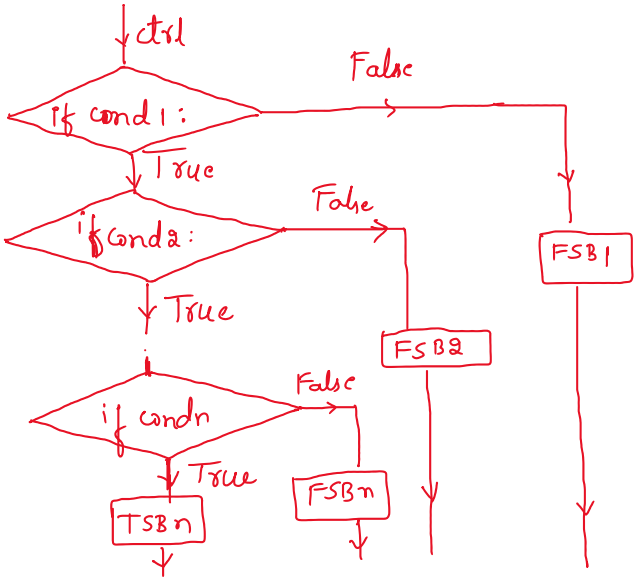
4) Nested if:

--- Whenever it is necessary to check a condition before another condition we use Nested if.
Or
Condition inside a condition is referred as Nested if.

Syntax:



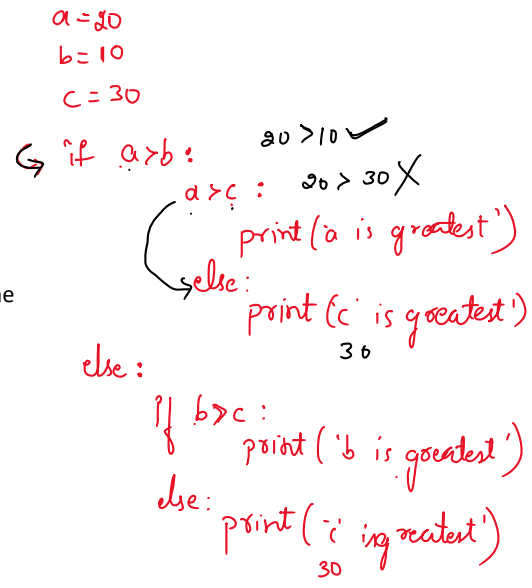
Flow diagram:



Programs:

Programs:

```
#Nested if

#WAP to check whether the given character is vowel or consonant.
'''
ch=input('Enter the character: ')
if 'a'<=ch<='z' or 'A'<=ch<='Z':
   if ch in 'aeiouAEIOU':
      print(ch,'is vowel')
   else:
      print(ch,'is consonant')
else:
   print(ch,'is not alphabet')'''

#WAP to login to the instagram by entering the proper username
and password.
'''
username='__seclusive__'
password='saku@123'
un=input('Enter your username: ')
if un == username:
   pw=input('Enter your password: ')
   if pw == password:
      print('Login Successful')
   else:
      print('Incorrect password')
else:
   print('Invalid username')'''
```




```
# WAP to find the greatest among three numbers.
'''
a,b,c=int(input('Enter the num1: ')),int(input('Enter the num2: ')),int(input('Enter the num3: '))
if a>b:
   if a>c:
      print(a,'is greater')
   else:
      print(c,'is greater')
else:
   if b>c:
      print(b,'is greater')
   else:
      print(c,'is greater')'''
```

Assignment:

```
# WAP to find the greatest among four numbers.
```

Day-21

Looping Statement:

--- It is also used to control the flow of execution by executing the same set of instructions again and again.

Types:

- **While loop**
- **For loop**

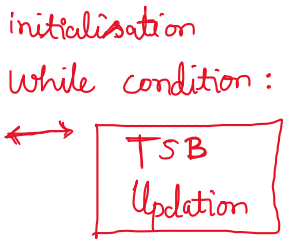
1) While loop:

--- It is used to execute the same set of instructions again and again until the condition become False.

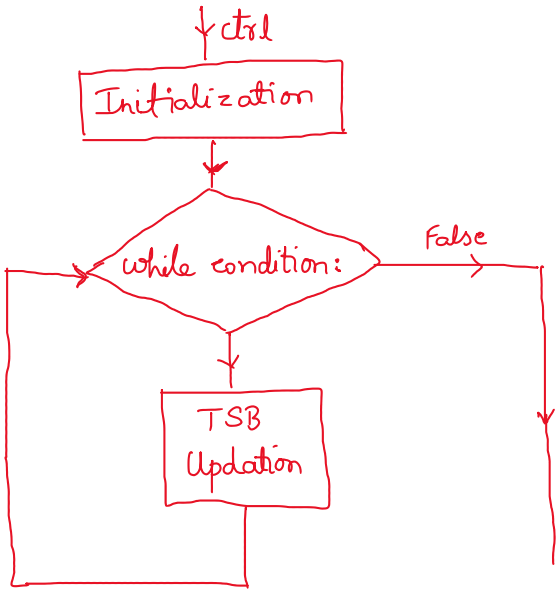
Note: There are 2 manditory / compulsory things in while loop

- Initialisation
- Updation

Syntax:

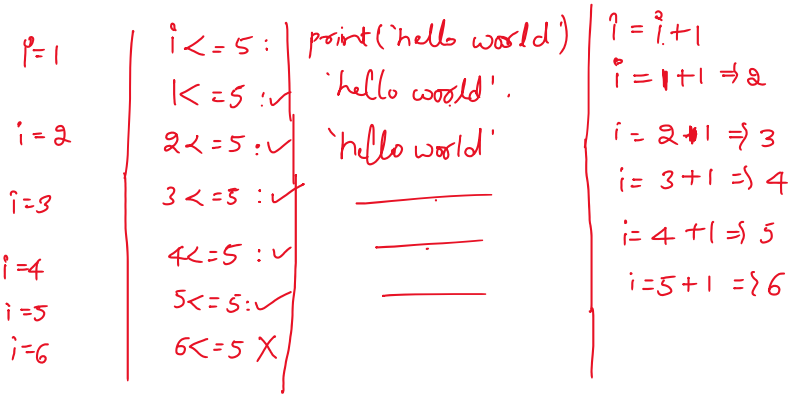


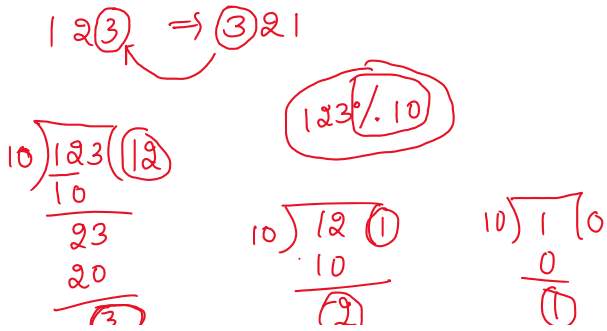
Flow Diagram:



Programs:

```
# While loop

#WAP to print hello world for 5 times.
'''
print('hello world')
print('hello world')
print('hello world')
print('hello world')
print('hello world')'''

''''
i=1
while i<=5:
   print('hello world')
   i=i+1'''


#WAP to print the n natural numbers
'''
num = int(input('Enter the number: '))
i=1
while i<=num:
   print(i)
   i+=1'''

# WAP to print the even numbers from 1 to 30
'''
num = int(input('Enter the number: '))
i=1
while i<=num:
   if i%2==0:
      print(i)
   i+=1'''

'''
num = int(input('Enter the number: '))
i=2
```





```
'''
num = int(input('Enter the number: '))
i=2
while i<=num:
   print(i)
   i=i+2'''

# WAP to print the number from n to 1.
'''
num = int(input('Enter the number: '))
i=num
while i>=1:
   print(i)
   i=i-1'''


# WAP to reverse the given number without typecasting.
'''
n = int(input('Enter the number: '))
res=0
while n>0:
   rem = n%10
   res = res + rem
   n = n//10
print(res)'''

#WAP to print the sum of n natural numbers.
'''
n = int(input('Enter the number: '))
out = 0
i=0
while i<=n:
    out = out + i
    i = i + 1
print(out)'''

#WAP to print the product of n natural numbers.
'''
n = int(input('Enter the number: '))
out = 1
i=1
while i<=n:
    out = out * i
    i = i + 1
print(out)'''

# WAP to print the every single character in a string.
'''
s = input('Enter the string: ')
i=0
while i< len(s):
   print(s[i])
   i+=1'''
```

23 → 20 → 3 ; 10 → 2 ; 0 → 1

321

res = res*10 + rem
= 0*10 + 3

res = 3

res = res + rem
= 0+3
res = 3
= 3+2 ⇒ 5

res = res*10 + rem
= 3*10 + 2
= 30+2
res = 32

res = res*10 + rem
= 32*10 + 1
= 320 + 1
= 321

```
n=int(input())
out=0
i=1
while i<=n
    out=out+i
    i=i+1
print(out)
```

3 ≤ 10 ✓
out = out + i
out = 0+1 ⇒ 1
⇒ 1+2 ⇒ 3
3+3 ⇒ 6

i = 1
i = 1+1 ⇒ 2
⇒ 2+1 ⇒ 3
3+1 ⇒ 4

Day-23

Continuation of while loop programs,

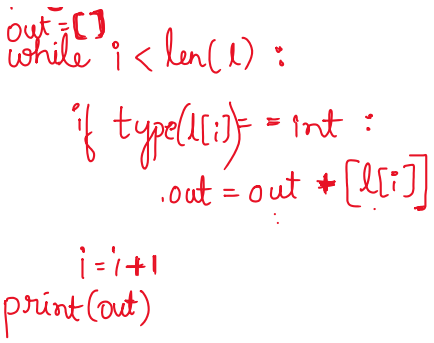
WAP to extract the integers from the list.

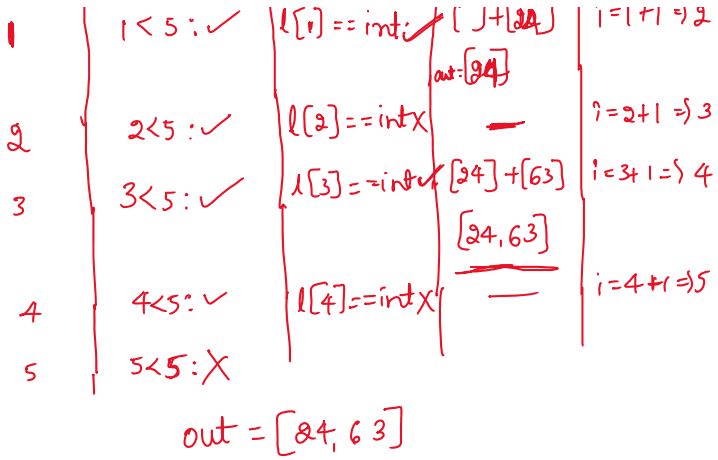
l = [2.3, 24, 7+9j, 63, 'hello']
0 1 2 3 4

i = 0
out = []
while i < len(l):

len(l) = 5

	i< len(l):	type(i)==int	out = out+[l[i]]	i = i+1
0	0<5: ✓	l[0]==int ✗	—	i = 0+1 ⇒ 1
1	1<5: ✓	l[1]==int: ✓	[ ]+[24]	i = 1+1 ⇒ 2
			out: [24]	





Practical proof:

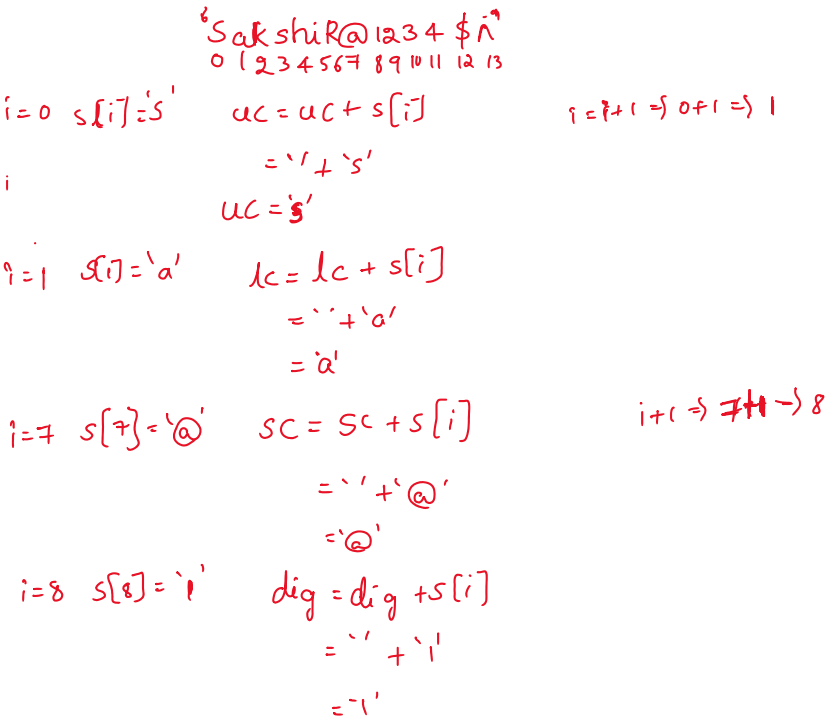
```
l = eval(input('Enter the list: '))
i = 0
out = []
while i < len(l):
   if type(l[i]) == int:
      out = out + [l[i]]
   i = i + 1
print(out)
```

Or

```
l = eval(input('Enter the list: '))
i = 0
out = []
while i < len(l):
   if type(l[i]) == int:
      out.append(l[i])
   i = i + 1
print(out)
```

WAP to extract uppercase, lowercase, digits and special characters seperately into 4 different variables.

```
s  = input('Enter the string: ')
uc = ''
lc = ''
dig = ''
sc = ''
i = 0
while i < len(s):
   if 'A'<=s[i]<='Z':
      uc = uc + s[i]
   elif 'a'<=s[i]<='z':
      lc = lc + s[i]
   elif '0'<=s[i]<='9':
      dig = dig + s[i]
   else:
      sc = sc + s[i]
   i = i + 1
print(uc)
print(lc)
print(dig)
print(sc)
```



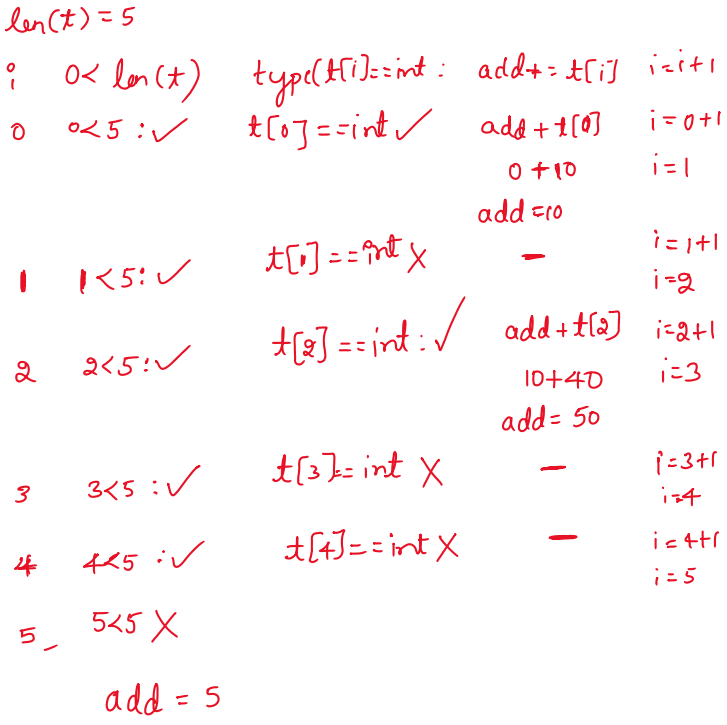
Output:

```
Enter the string: SakshiR@1234$^
SR
```

akshi
1234
@$^

WAP to find the sum of integers in the tuple.

```
t = eval(input('Enter the tuple: '))
add = 0
i = 0
while i < len(t):
   if type(t[i]) == int:
      add = add + t[i]
   i = i + 1
print(add)
```



Output:

```
Enter the tuple: (10,2.3,40,True,'bye')
50
```

From Uppercase to lowercase conversion:

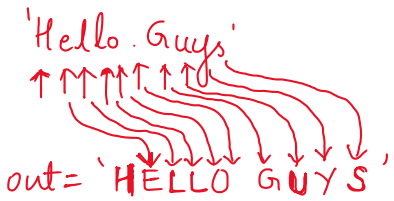
```
ord('A')
65
ord('a')
97
chr(97)
'a'
chr(ord('A')+32)
'a'
chr(ord('A')+32)
'a'
chr(ord('B')+32)
'b'
chr(ord('C')+32)
'c'
```

From Lowercase to Uppercase conversion:

```
ord('A')
65
ord('a')
97
chr(ord('a')-32)
'A'
chr(ord('b')-32)
'B'
chr(ord('c')-32)
'C'
```

WAP to convert all the lowercase alphabet from the string to uppercase alphabet.

```
s=input('Enter the string: ')
out = ''
i=0
while i<len(s):
   if 'a'<=s[i]<='z':
      out = out + chr(ord(s[i])-32)
   else:
      out = out + s[i]
   i = i + 1
print(out)
```



Output:

```
Enter the string: Hello Guys
HELLO GUYS
```

Toggle:
--- Converting from lowercase to uppercase and uppercase to lowercase and keeping the rest of the characters as it is in the string.

WAP to toggle the string.

```
s=input('Enter the string: ')
toggle = ''
i=0
while i<len(s):
   if 'a'<=s[i]<='z':
      toggle = toggle + chr(ord(s[i])-32)
   elif 'A'<=s[i]<='Z':
      toggle = toggle + chr(ord(s[i])+32)
   else:
      toggle = toggle + s[i]
   i = i + 1
print(toggle)
```

Output:

```
Enter the string: Python Is The Champion 21*^$^#$
pYTHON iS tHE cHAMPION 21*^$^#$
```

WAP to find the product of all the float numbers present at the odd index in a given list.

```
l = eval(input('Enter the list: '))
prod = 1
i = 0
while i<len(l):
   if type(l[i])==float and i%2!=0:
      prod = prod * l[i]
   i = i + 1
print(prod)
```

Output:

```
Enter the list: [2.3,4.6,7.1,6.2,8.5]
28.52
4.6 * 6.2
28.52

Enter the list: [True,4.6,7.1,6.2,2+3j,'gm',2.6]
28.52
```

Day-24

For loop:

--- It is self-iterative loop.

Drawback of while loop :

- It requires index position of the values in collection to traverse but set and dictionary won't support for indexing.
- Initialization , condition and updation are manditory in while loop.

Advantage of for loop over while loop:

- No need of Initialization and updation.
- It can be used of all the collection datatypes.

range(): It is used to create the sequence of integers between the given values.

Syntax:

range(SV,EV+1,updation)
range(SV,EV- 1,updation)

Range shortcuts:

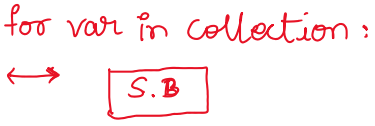
- If updation == +1
  range(SV, EV+- 1)

- If SV == 0 and updation == +1
  range(EV+1)

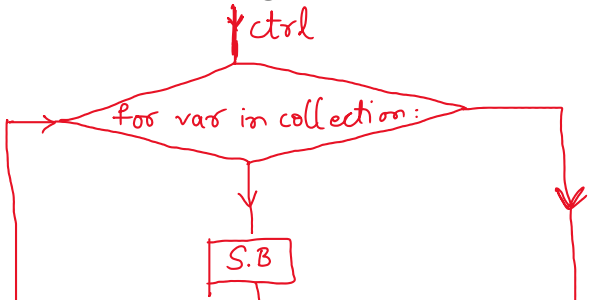
Key points -- To print the range of values use typecasting.

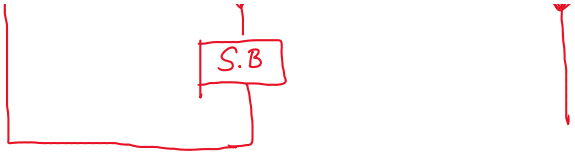
```
range(1,10+1)
range(1, 11)
list[range(1, 11)]
list[range(1, 11)]
list(range(1, 11))
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
```

Syntax:

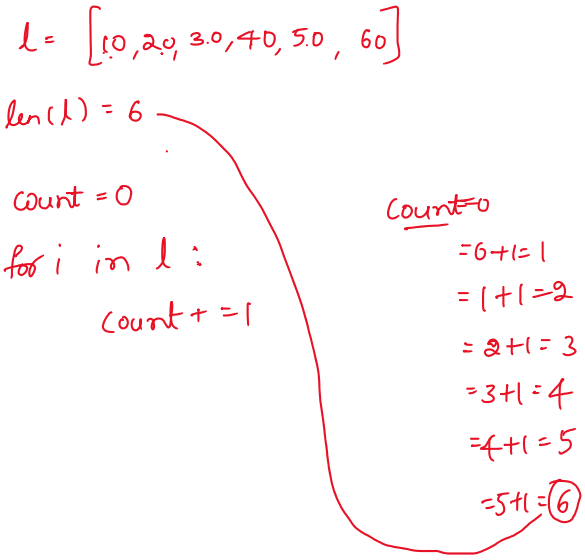


Flow diagram:

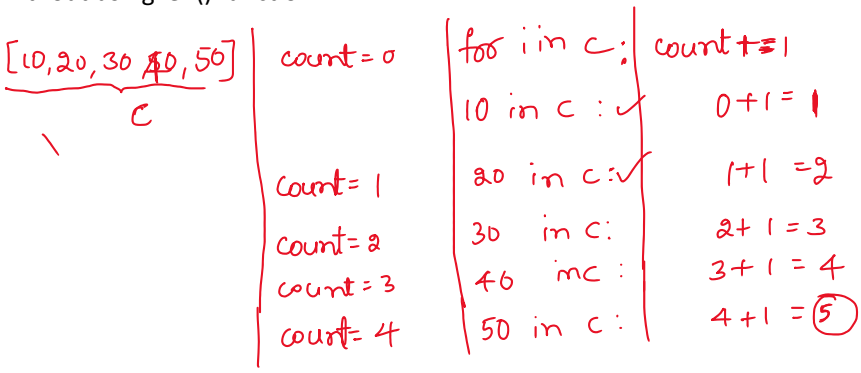




Programs:

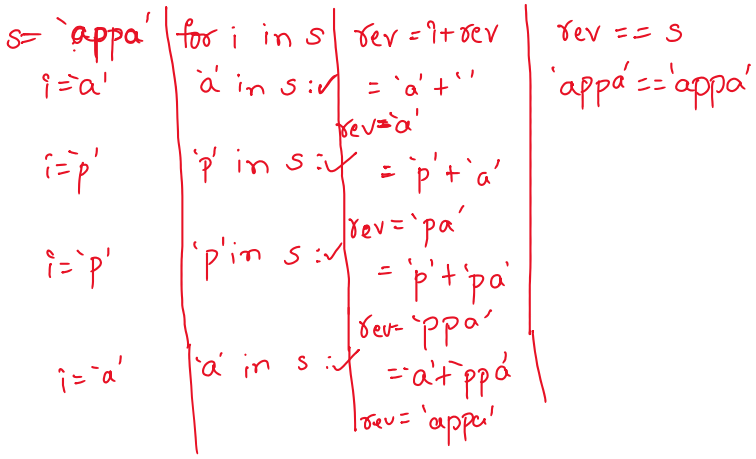



```
'''
l = [10,2.3,6+7j,78]
for i in l:
   print(i)'''

'''
for i in {1,7,4.5,45,True}:
   print(i)'''
'''
for i in {'a':10,'b':20,'c':30}:
   print(i)'''

'''
for i in range(1,7,2):
   print(i)'''

# Actual programs on for loop

# To find the length of the given collection without using len() function.

c = eval(input('Enter the collection: '))
count = 0
for i in c:
   count += 1
print(count)
```




```
# WAP to extract vowels from the string.
'''
s = input('Enter the string: ')
out = ''
i = 0
while i<len(s):
   if s[i] in 'aeiouAEIOU':
      out+=s[i]
   i+=1
print(out)'''

'''
s = input('Enter the string: ')
out = ''
for i in s:
   if i in 'aeiouAEIOU':
      out+=i
print(out)'''

# WAP to replace space by an underscore in a given string.
'''
s = input('Enter the string: ')
rep = ''
for i in s:
   if i == ' ':
```

```
        rep += '_'
    else:
        rep += i
print(rep)'''

# WAP to check whether the string is palindrome or not without using slicing.

s = input('Enter the string: ')
rev = ''
for i in s:
    rev = i + rev
print(rev)
if rev == s:
    print('palindrome')
else:
    print('not palindrome')
```




```
# WAP to remove duplicates values from the list.
'''
l = eval(input('Enter the list: '))
out = []
for i in l:
    if i not in out:
        out.append(i)
print(out)'''

# Get the following output
'''
 Input :     (12,3.4,'hello',8+9j,'python','bye',2.6,True)
 Output :   {'hello':5,'python':6,'bye':3}'''

'''
t = eval(input('Enter the tuple: '))
out = {}
for i in t:
    if type(i)==str:
        out[i]= len(i)
print(out)'''

# Get the following output.
'''
Input : [12,'data',3.4,True,'science',3+7j,'engineer']
Output : {'data':'da','science':'se','engineer':'er'}'''

'''
l = eval(input('Enter the list: '))
out = {}
for i in l:
    if type(i) == str:
        out[i]=i[0]+i[-1]
print(out)'''

# WAP to get the following output
'''
Input : 'ApPlE#23'
Output : {'A':a,'p':P,'P':p,'l':'L','E':e}'''

'''
s = input('Enter the string: ')
```

```
out = {}
for i in s:
   if 'A'<=i<='Z':
      out[i] = chr(ord(i)+32)
   elif 'a'<=i<='z':
      out[i] = chr(ord(i)-32)
print(out)'''

# WAP to get the following output
'''
Input : 'hai hello how are you'
Output : {'hai':3,'hello':5,'how':3,'are':3,'you':3}'''

'''
s = input('Enter the string: ')
out = {}
a = s.split()
for i in a:
   out[i] = len(i)
print(out)'''

# WAP to get the following output
'''
Input : 'hello fellow coders'
Output : 'olleh wollef sredoc'          '''

'''
s = input('Enter the string: ')
out = []
a = s.split()
for i in a:
   out.append(i[::-1])
print(''.join(out)) '''

# WAP to get the following output
'''
Input : 'abcabacbcc'
Output : 'a3b3c4'        '''

'''
s = input('Enter the string: ')
out = ''
for i in s:
   if i not in out:
      c = s.count(i)
      out += i + str(c)
print(out)'''
```

⇒ Tracing

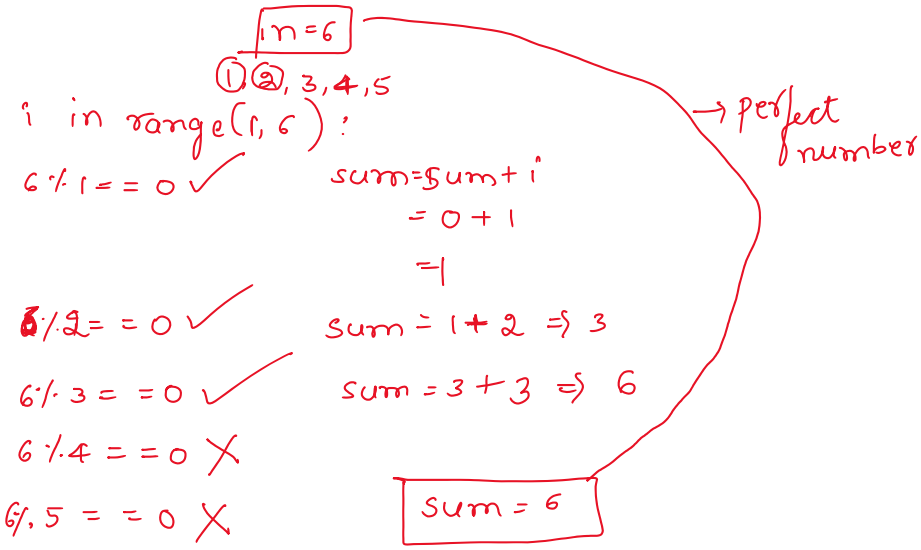
s = 'abcabacbcc'

i in s	if i not in out:	c = s.count(i)	out += i + str(c)
i = a	a not in '' ✓	c = 3	= 'a' + '3'
			out = 'a3'
i = b	b not in 'a3' ✓	c = 3	= 'b' + '3'
			= 'a3' + 'b3'
			= 'a3b3'
i = c	c not in 'a3b3' ✓	c = 4	= 'a3b3' + 'c' + '4'
i = a	a not in 'a3b3c4' ✗		= 'a3b3c4'

out = 'a3b3c4'

What is perfect number?
--- If the sum of all the factors of a given number (excluding the number itself) becomes equal to the given number means , we can say the number is Perfect number.

```
# WAP to check whether the given number is perfect number or not.
'''
n = int(input('Enter the number: '))
sum = 0
for i in range(1,n):
   if n % i == 0:
      sum += i
if sum == n:
   print('perfect number')
else:
   print('not perfect number')'''
```



Day-25

Armstrong Number : We have to power each and every digit of the number by its length and add the value to a variable. If the added value becomes equal to the number itself means the number is armstrong number.

153 => 1**3 + 5**3 + 3**3 => 1+125+27 => 153

s='153'
len(s)=3

153 == 153

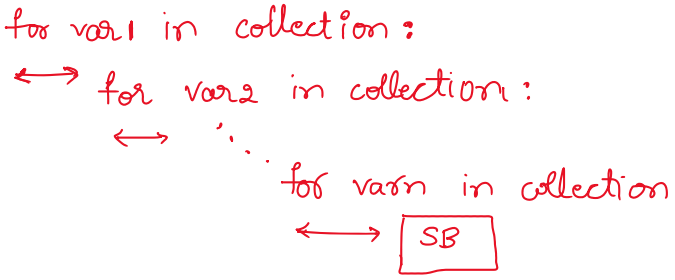
```
# WAP to find the given integer is armstrong number of not.
'''
n = int(input('Enter the number: '))
sum = 0
a = str(n)
for i in a:
   sum += int(i) ** len(a)
if sum == n:
   print('armstrong number')
else:
   print('not armstrong number')'''

# WAP to extract key-value pair from the dictionary only if key is of string type.
'''
d = eval(input('Enter the dictionary: '))
out = {}
for i in d:
   if type(i) == str:
      out[i] = d[i]
print(out)'''

# Get the following output
'''
Input : { 'M155': 'morning batch',   'E140':'Evening batch',    'M9':'weekend batch'}
output : { 'morning batch': 'M155',  'Evening batch':'E140',   'weekend batch':'M9'} '''

'''
d = eval(input('Enter the dictionary: '))
```

```
out = {}
for i in d:
   out[d[i]]=i
print(out)'''
```

Nested for loop:

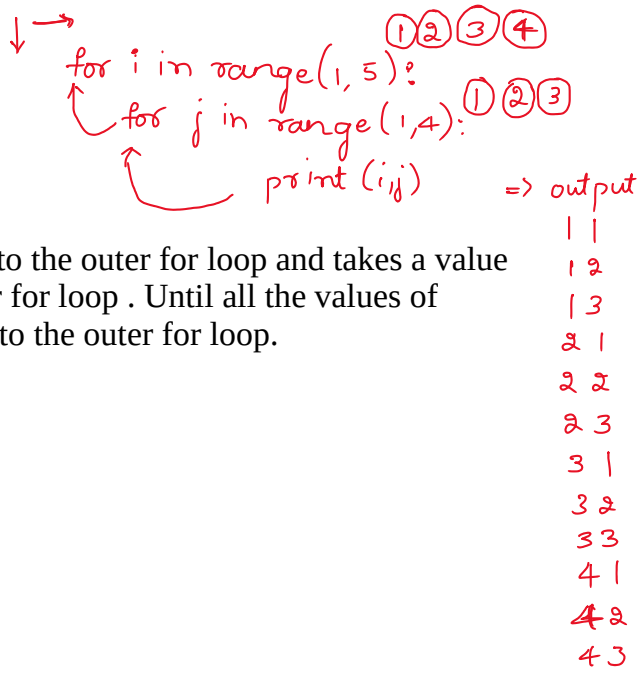
--- It is a phenomenon where we write a for loop inside another for loop.

Syntax:



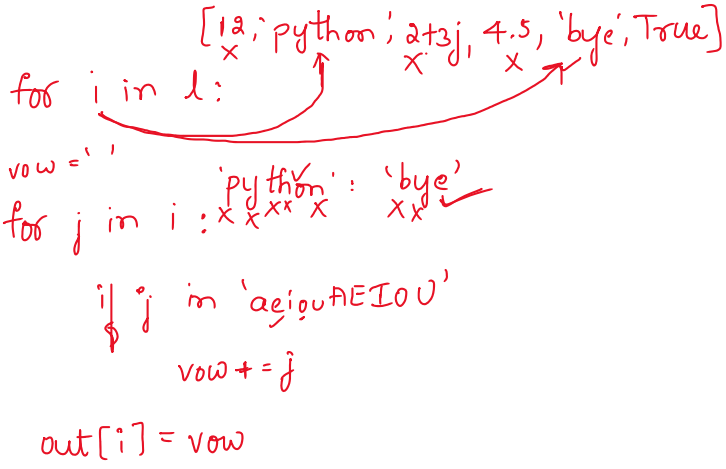
Programs

```
for i in range(1,5):
   for j in range(1,4):
      print(i,j)
```



Reason: First the variable will comes to the outer for loop and takes a value From the collection and enters to inner for loop . Until all the values of Inner for loop ends it will not go back to the outer for loop.

```
# Guess th following output.
'''
Input :  [12,'python',2+3j,4.5,'bye',True]
Output : {'python':'o','bye':'e'}    '''

'''
l = eval(input('Enter the list: '))
out = {}
for i in l:
   if type(i) == str:
      vow=''
      for j in i:
         if j in 'aeiouAEIOU':
            vow += j
      out[i] = vow
print(out)'''
```




```
# Guess the following output.
```

```
'''
Input :  [12,'python',2+3j,4.5,'bye',True]
Output : {'python':'pythn','bye':'by'}   '''

'''
l = eval(input('Enter the list: '))
out = {}
for i in l:
   if type(i) == str:
      cons=''
      for j in i:
         if j not in 'aeiouAEIOU':
            cons += j
      out[i] = cons
print(out)'''


# Guess the following output.
'''
Input :  [12,'python',2+3j,4.5,'bye',True]
Output : {'python':'PYTHON','bye':'BYE'}   '''
'''
l = eval(input('Enter the list: '))
out = {}
for i in l:
   if type(i) == str:
      upper=''
      for j in i:
         if 'a'<=j<='z':
            upper += chr(ord(j)-32)
         else:
            upper += j
      out[i] = upper
print(out) '''
```

Day-26

Strong Number: If the number is equal to the sum of the factorial of individual digits, the we call it as Strong number.

145 = 1! + 4! + 5!

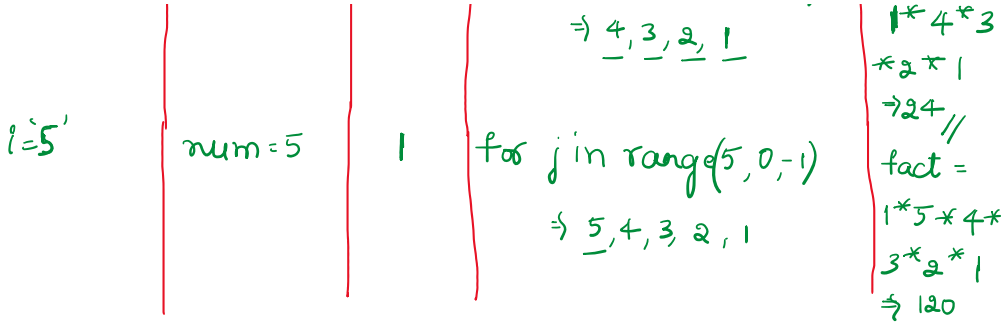
```
# WAP to check whether the given number is strong number or not
'''
n =  int(input('Enter the number: '))
sum_fact = 0
a = str(n)
for i in a:
   num = int(i)
   fact = 1
   for j in range(num,0,-1):
      fact *= j
   sum_fact += fact
if sum_fact == n:
   print('Strong Number')
else:
   print('Not strong number') '''
```

n ⇒ 145

a = str(n) = '145'

for i in a:	num = int(i)	fact = 1	for i in range(num, 0, -1)	fact* = ∫
i = '1'	num = 1	1	for i in range(1, 0, -1) ⇒ (1)	fact = 1*1 = 1
i = '4'	num = 4	1	for i in range(4, 0, -1) ⇒ 4, 3, 2, 1	fact = 1*4*3*2*1

```
print('Not strong number') '''
```



sum_fact += fact

= 1 + 24 + 120

sum_fact = 145 = = n

Patterns:

--- Using Nested for loop we print patterns.

Programs.

```
# Patterns
'''
* * *                    '''

'''
for i in range(1,4):
   print('*',end=' ') '''

'''
* * *
* * *
* * *                          '''

for i in range(1,4):
   for j in range(1,4):
      print('*', end=' ')
   print()
```